Chapter 1

Introduction to Logic Basic

1.1 Presentation

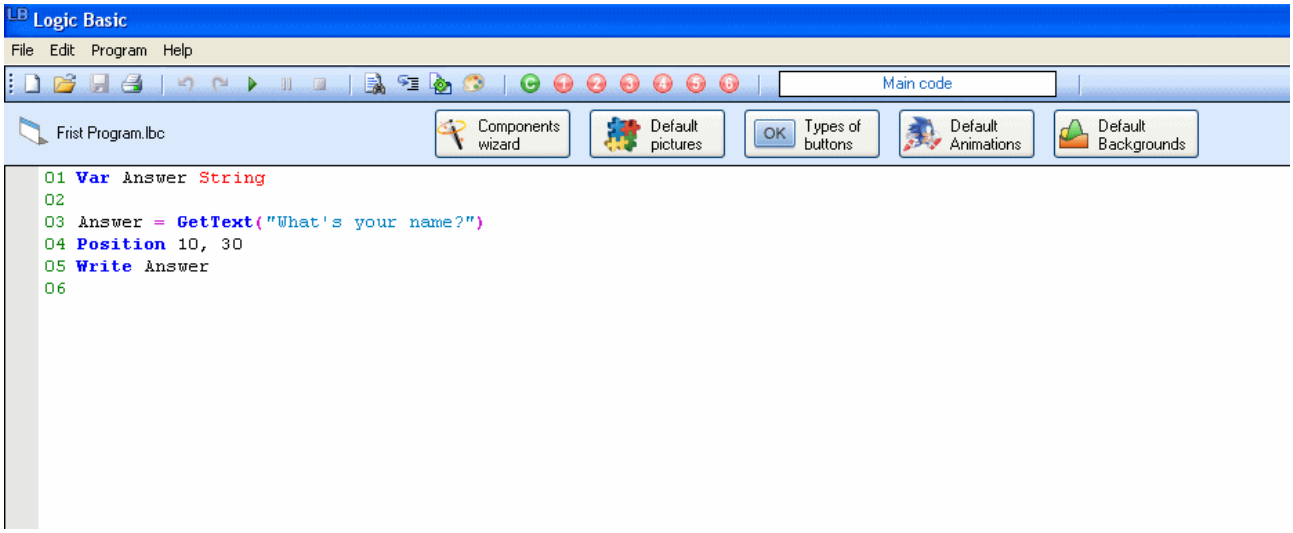
Logic Basic is a high-level programming language, inspired by the older versions of the Basic language, such as MS Basic Compiler, Microsoft QuickBasic, GW-Basic, and so on. Basic stands “Beginner's All-Purpose Symbolic Instruction Code”.

One of the purposes of Logic Basic is to provide programming in Basic adapted to modern window operating systems, since the older versions of Basic were designed for the MS-DOS operating system. Another purpose is to facilitate program development by providing simple commands and functions that perform tasks that are difficult to implement in other programming languages, making programming pleasant, simple, fun and interesting.

Logic Basic allows the user to easily develop commercial programs, reports, animations, games, presentations, educational programs, musicals and more.

1.2 The Logic Basic Interface

Logic Basic has a code environment where you write your program, this environment has a text box to the main code and more 6 text boxes for the code extensions. It also has the environment of windows, where the results of the program will be shown.



In Logic Basic the program is executed sequentially, that is, from the first line to the last line, or until the line where the **EndProgram** command is placed. However, the order of program execution can be changed by the **GoTo** command or by subroutines and functions.

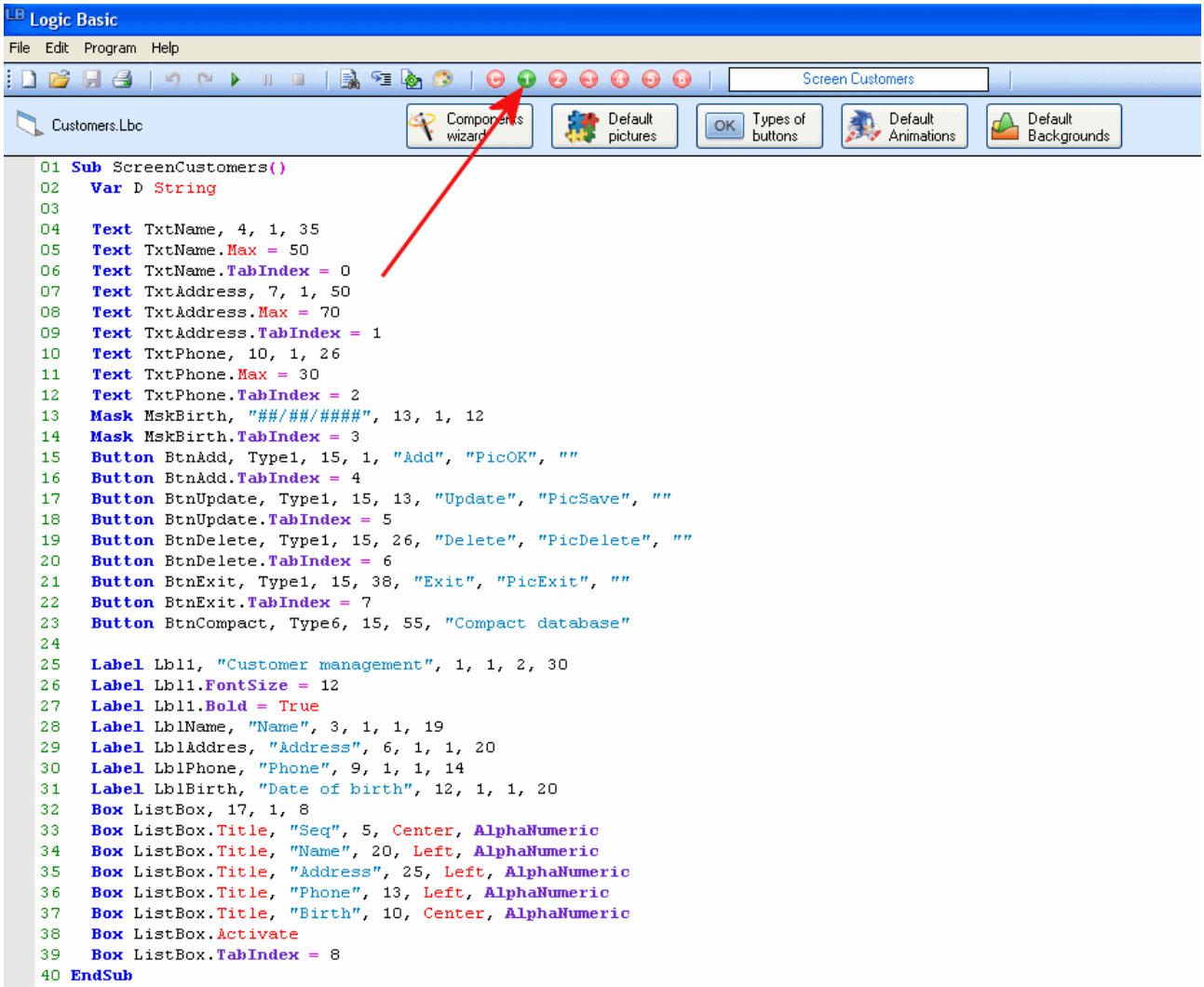
To run the program, click on the "Run" button (green arrow to the right) or press the F5 key, in this way Logic Basic will load the main window, which is the location that will receive the code commands, which can write texts, create buttons, text boxes, pictures, animations, background images, draw geometric shapes, and also allow you to interact with the keyboard, mouse, hard disk, microphone, and so on.

1.3 Code extensions

There are also "Code extensions", which are text boxes where you can put some code snippets to better organize your program. To access the main code, you must press the code button (button containing the letter "C"), and to access the extensions, simply press one of the 6 red buttons on the right side of the code button.

You can imagine these extensions as a continuation of the main code, so that when you run the program, it will be as if they were concatenated (stitched) to the main code.

If you press one of the buttons of "code extensions", the Logic Basic shows in the text box below the code that corresponds to the respective extension. At the top right of the code window there is a text box labeled "Code Name" where you can be written a name to identify the code extension. This name is optional, it is not necessary to put it, but it is recommended to improve the clarity and reasoning of the program.

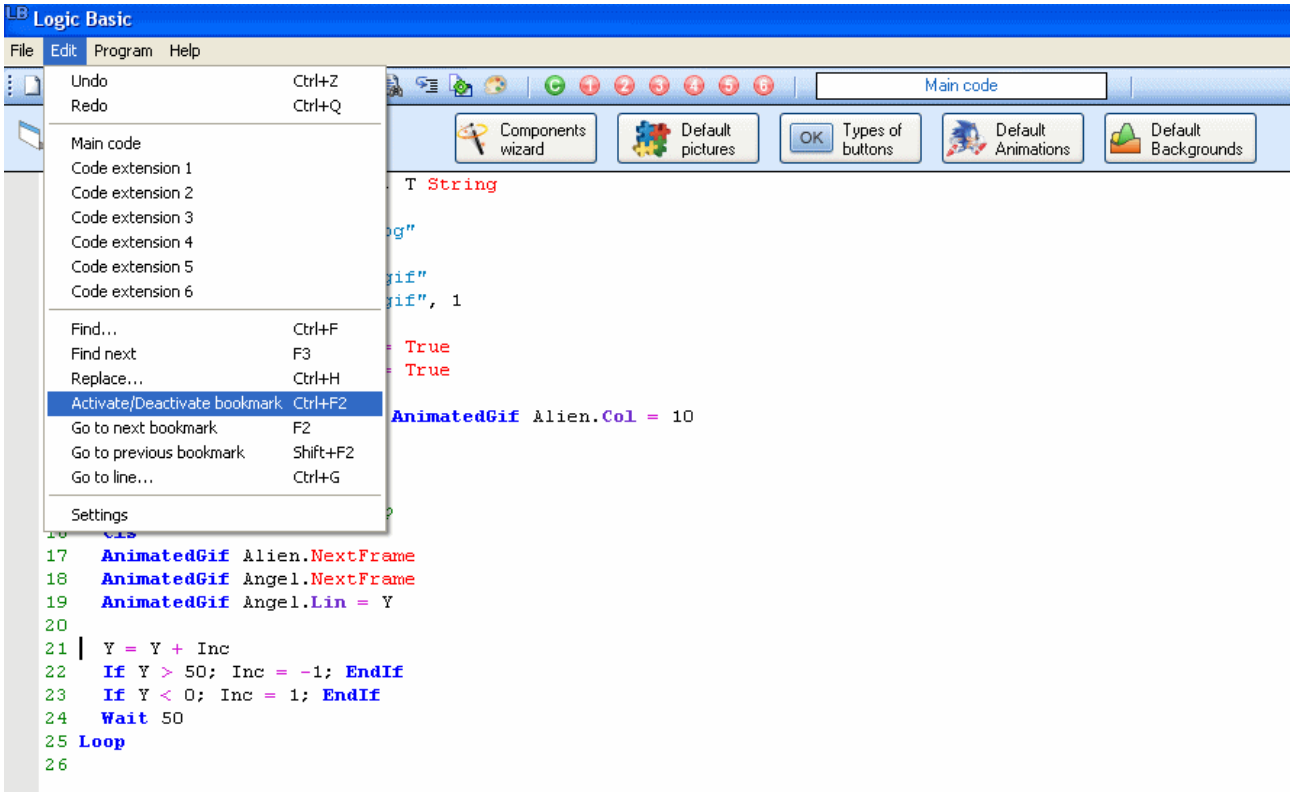


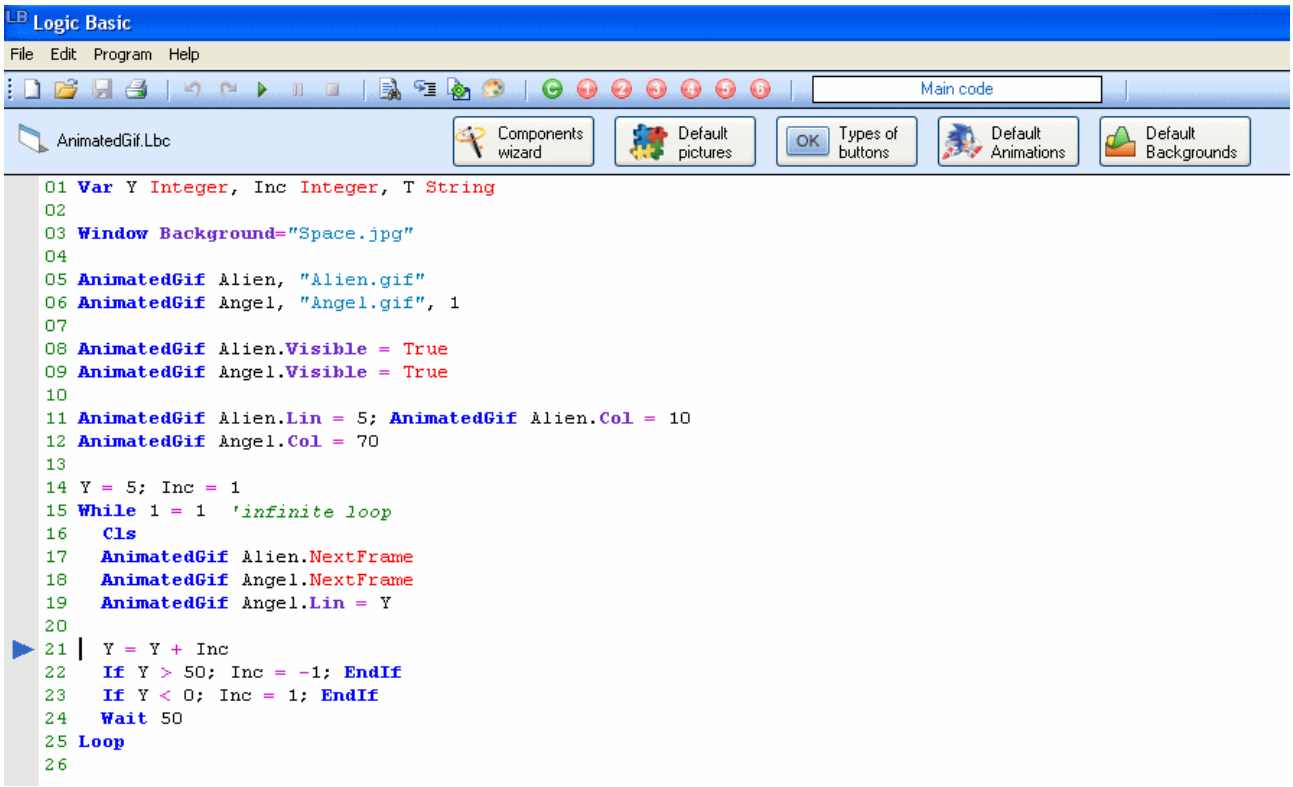

```
01 Sub ScreenCustomers()
02   Var D String
03
04   Text TxtName, 4, 1, 35
05   Text TxtName.Max = 50
06   Text TxtName.TabIndex = 0
07   Text TxtAddress, 7, 1, 50
08   Text TxtAddress.Max = 70
09   Text TxtAddress.TabIndex = 1
10   Text TxtPhone, 10, 1, 26
11   Text TxtPhone.Max = 30
12   Text TxtPhone.TabIndex = 2
13   Mask MskBirth, "##/##/####", 13, 1, 12
14   Mask MskBirth.TabIndex = 3
15   Button BtnAdd, Type1, 15, 1, "Add", "PicOK", ""
16   Button BtnAdd.TabIndex = 4
17   Button BtnUpdate, Type1, 15, 13, "Update", "PicSave", ""
18   Button BtnUpdate.TabIndex = 5
19   Button BtnDelete, Type1, 15, 26, "Delete", "PicDelete", ""
20   Button BtnDelete.TabIndex = 6
21   Button BtnExit, Type1, 15, 38, "Exit", "PicExit", ""
22   Button BtnExit.TabIndex = 7
23   Button BtnCompact, Type6, 15, 55, "Compact database"
24
25   Label Lbl1, "Customer management", 1, 1, 2, 30
26   Label Lbl1.FontSize = 12
27   Label Lbl1.Bold = True
28   Label LblName, "Name", 3, 1, 1, 19
29   Label LblAddres, "Address", 6, 1, 1, 20
30   Label LblPhone, "Phone", 9, 1, 1, 14
31   Label LblBirth, "Date of birth", 12, 1, 1, 20
32   Box ListBox, 17, 1, 8
33   Box ListBox.Title, "Seq", 5, Center, AlphaNumeric
34   Box ListBox.Title, "Name", 20, Left, AlphaNumeric
35   Box ListBox.Title, "Address", 25, Left, AlphaNumeric
36   Box ListBox.Title, "Phone", 13, Left, AlphaNumeric
37   Box ListBox.Title, "Birth", 10, Center, AlphaNumeric
38   Box ListBox.Activate
39   Box ListBox.TabIndex = 8
40 EndSub
```

1.4 Bookmarks

The LB Editor has the “Bookmark” feature that allows the user to mark certain lines of code and then find them quickly, which is useful when the code is too long. To mark or unmark a line, position the cursor on the line and press CTRL + F2. Therefore, when the user

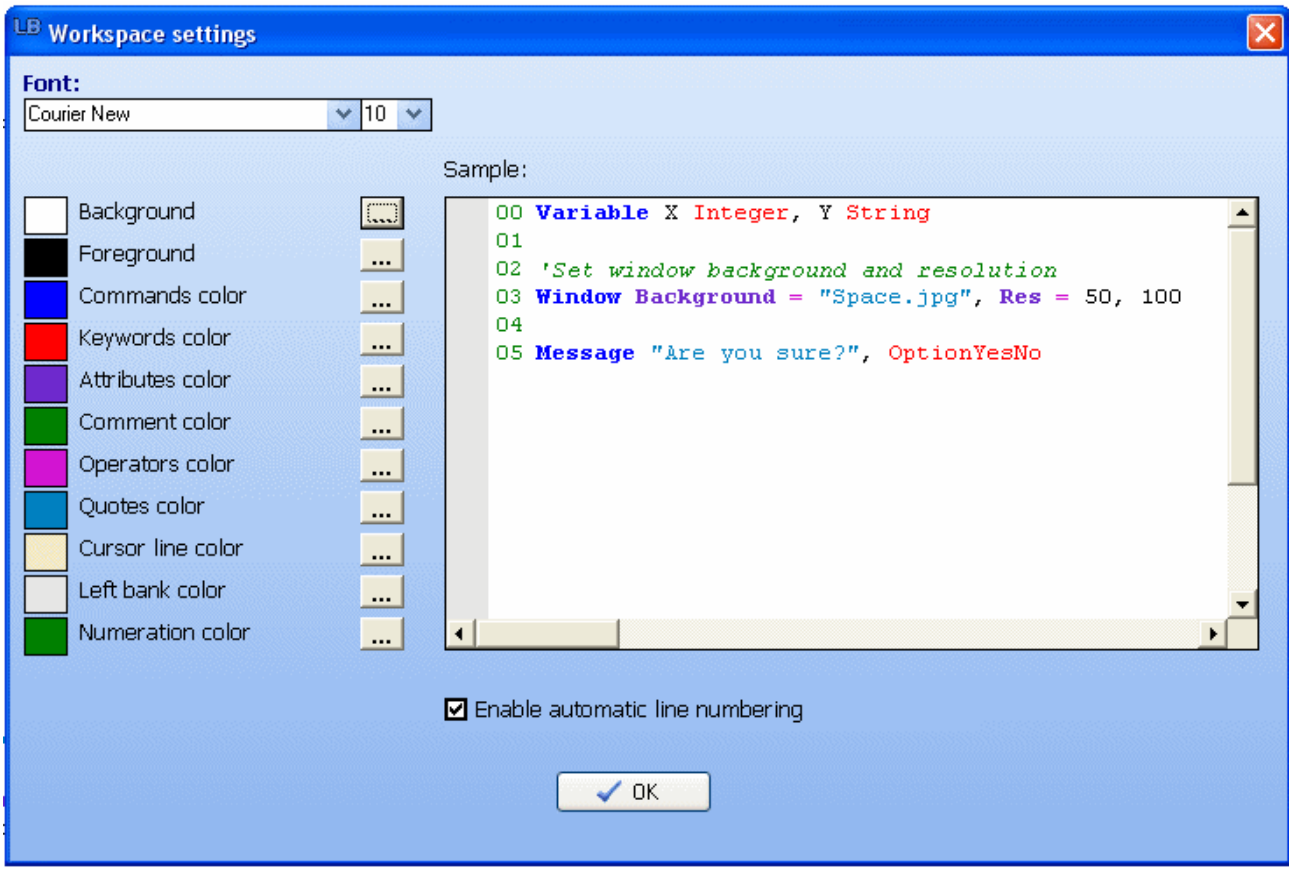
wishes to go to the next line containing a bookmark, simply press the F2 key.





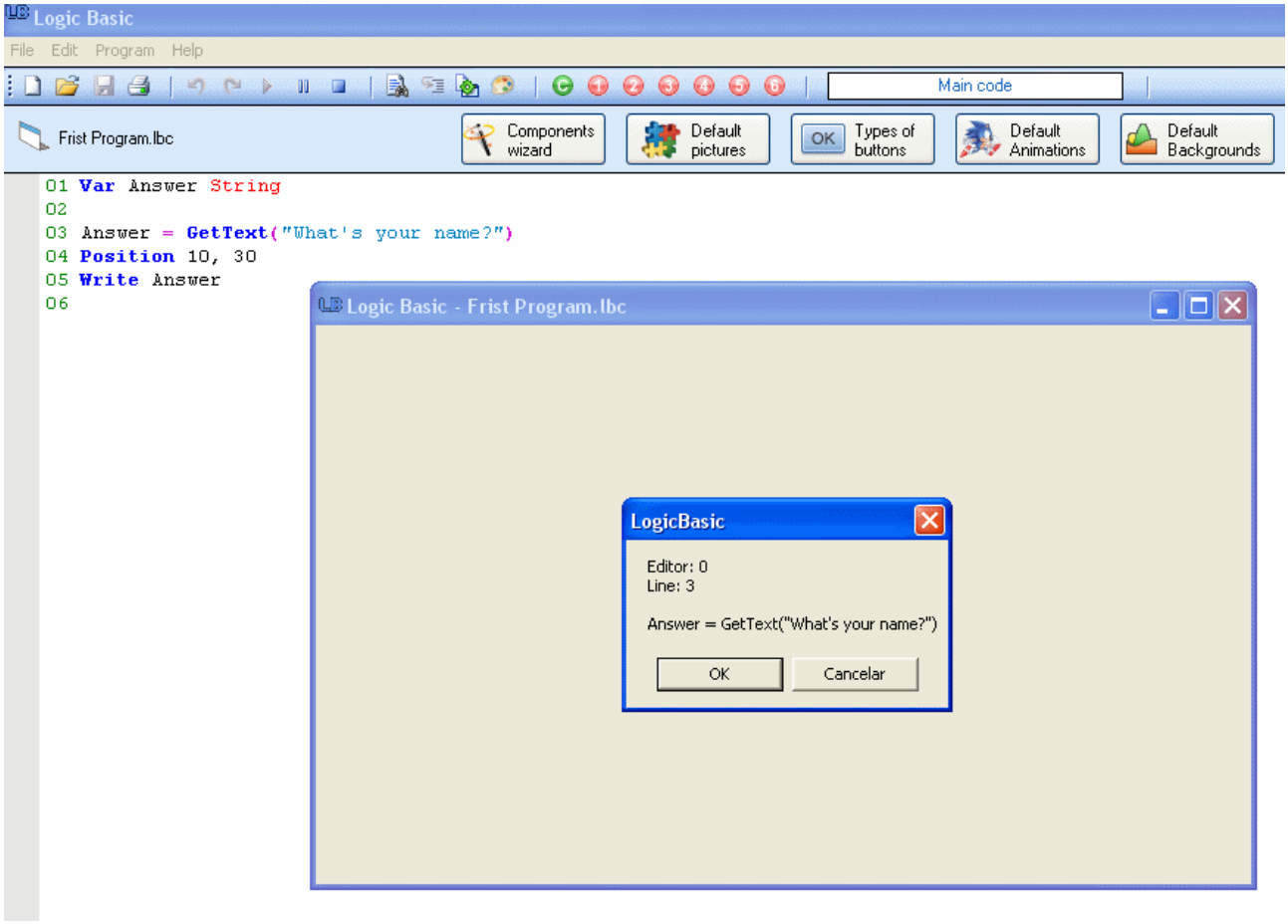
1.5 Workspace Settings

The code environment can be configured according to user preferences by accessing the **Edit → Settings** options, which will display a window that allows you to configure font, commands color, background color, and to turn lines numbering on or off.



1.6 Debug the program code

Often programming errors occur, undeclared or incorrectly declared variables, syntax errors, invalid characters in the code, and so on. To facilitate the identification of the cause of errors, a debugging routine was created, so that when executing the program, the LB will display in a small window the number of the editor (0 for main code, 1 for 6 for code extensions), the number line and the contents of the text being executed. Then the user can follow the execution of the code step by step to the line where the error is occurring.



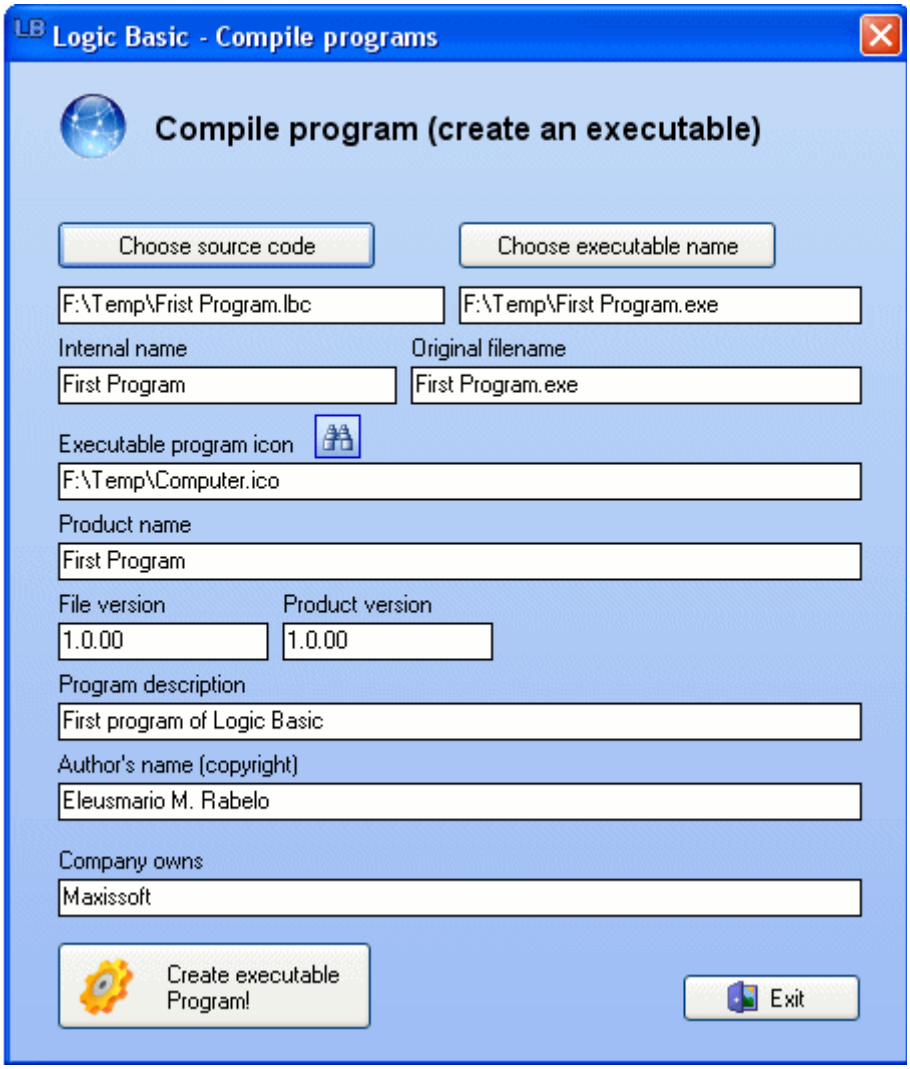
To start debugging the program code, go to the options **Program → Debug** or press the corresponding button on the top bar of the window.

1.7 Compile program (create an executable)

When you create a program, to run it, your source code must be in the text editor of Logic Basic, so anyone can view and change their content.

But let's assume you want to distribute or sell your program without people having access to your source code. For this you can create an auto-executable program (with the extension .exe), so that people can run it without needing Logic Basic.

To compile your program, go to the **Program → Compile** options or click the corresponding button in the upper bar of the window:



To compile the program follow these steps:

- Select the name of the source code (with extension .LBC, abbreviation of Logic Basic Code), you can enter the name or click on the "Choose source code" button;
- Choose the name of the executable program (with .exe extension), you can enter the name or click the "Choose executable name" button;

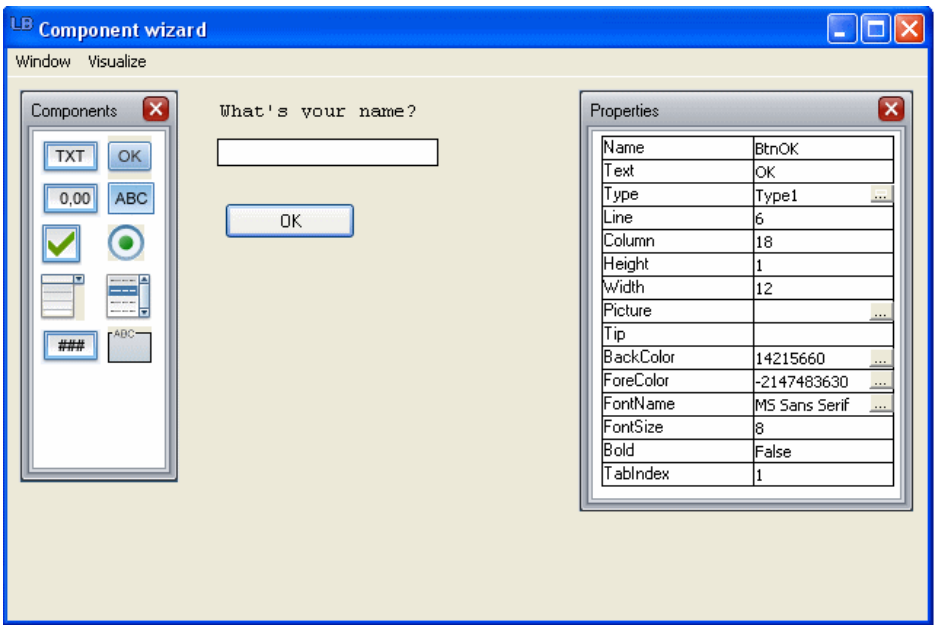
The following steps are optional:

- Internal name: Original name of the file without the extension.
- Original filename: Name of the file with the extension.
- Executable program icon: File name of the program icon (with .ico extension), enter the path of the icon or click on the binocular to find it.
- Product name: The name of the product this file is distributed with.
- File version: The version number of the file.
- Product version: The version of the product this file is distributed with.
- Program description: Brief description of the program.
- Author's name (copyright): Name of the author of the program.
- Company owns: Name of the company owner of the program.

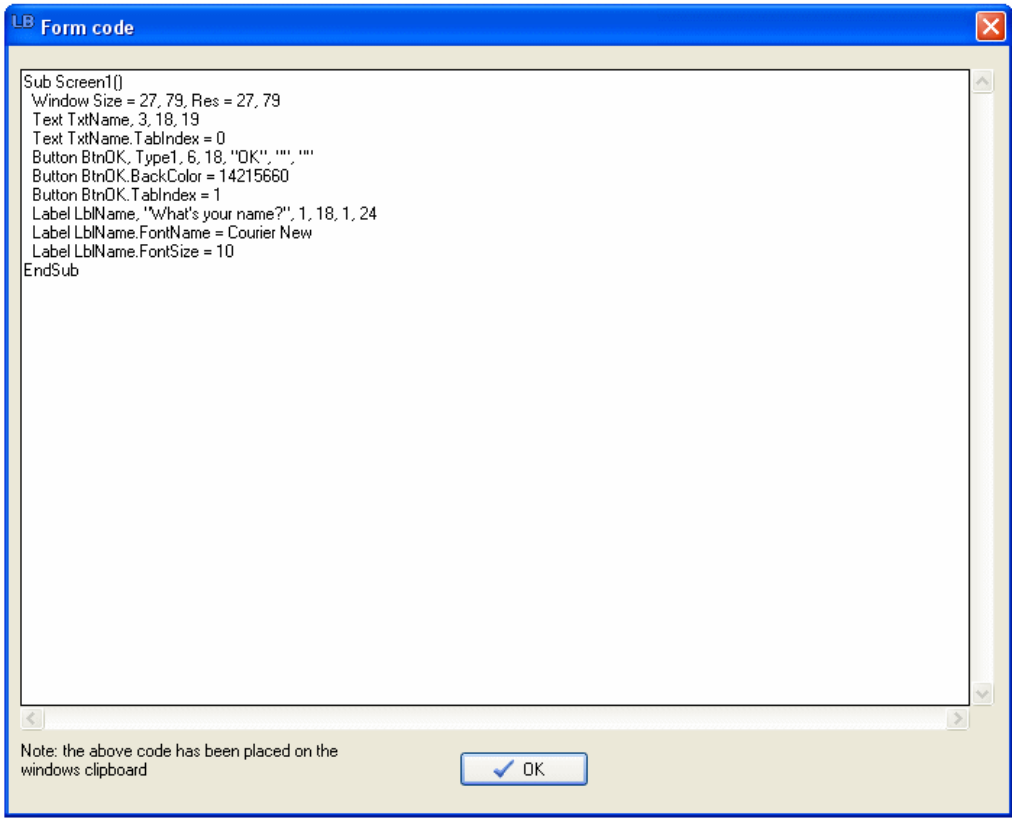
1.8 Components Wizard

Logic Basic provides a tool to make it easier for you to create components and design the interface of your application.

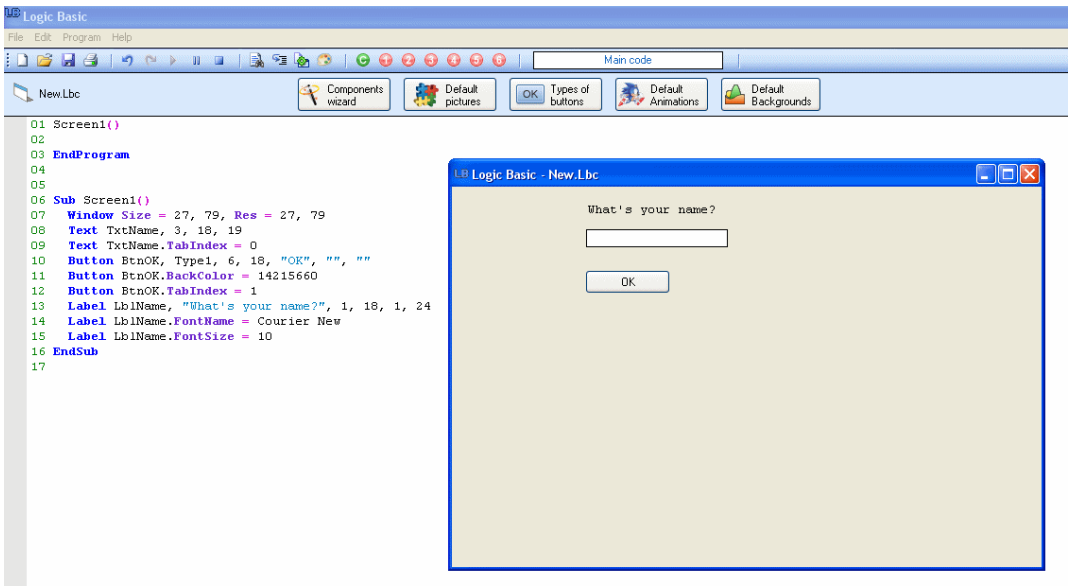
When you access this routine, you will see a window with a small toolbox containing several types of components: Text, Button, Currency, Check Button, Radio Button, Combo Box, List Box, Mask, and Frame. Click on a component and then mark in the window the location that you want to place it, for example:



When the design of your page is ready, you can generate the corresponding code by clicking the **Window → Generate Code** options or pressing the F3 key, which will then display a text box with the code that you can copy and paste in your program:

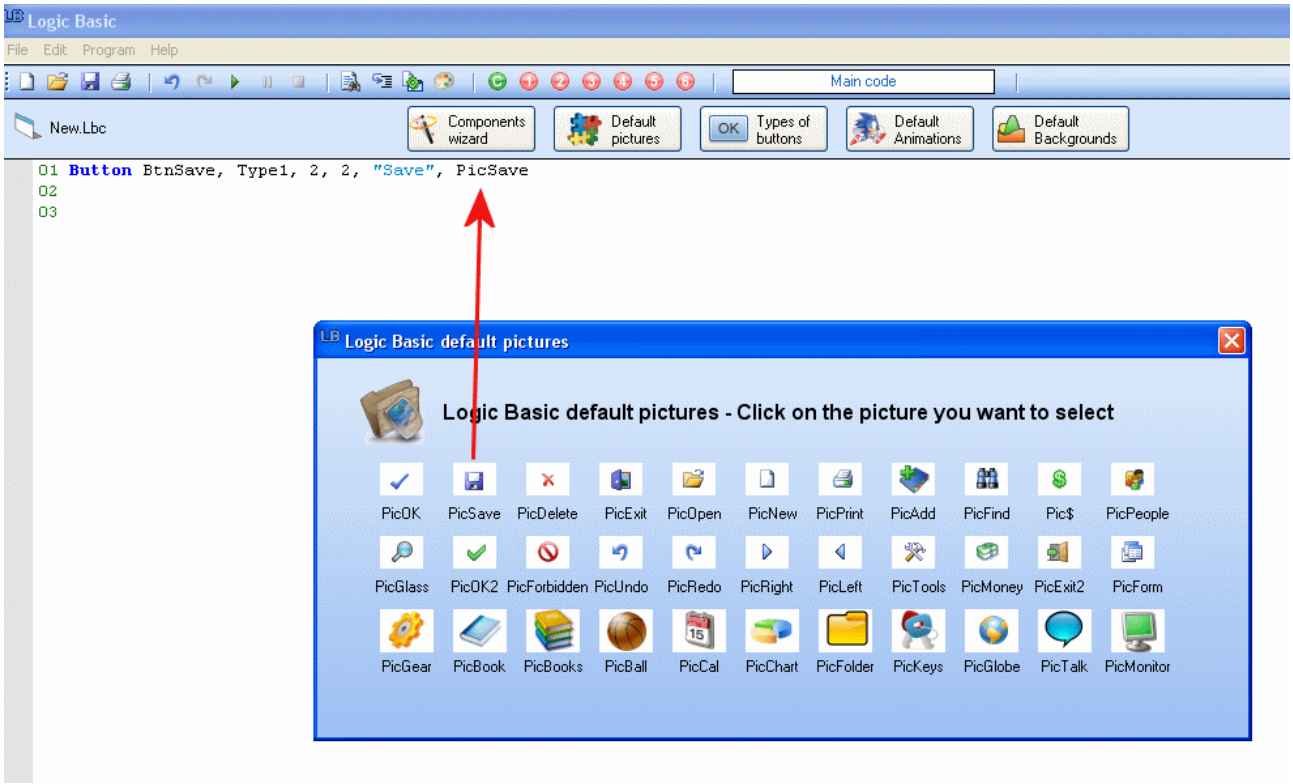


Note that the generated code is a subroutine, so you should place it after the **EndProgram** command:



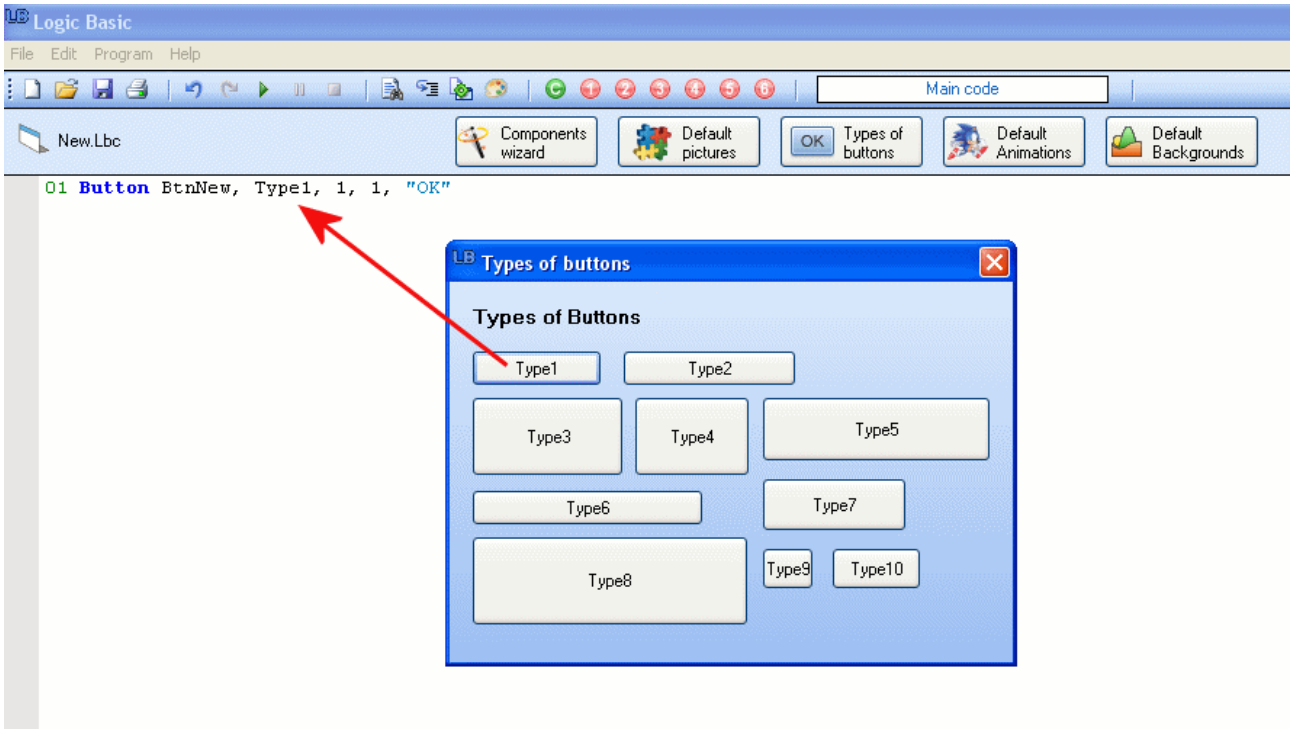
1.9 Default Pictures

Logic Basic provides a window with several default pictures, whose names can be assigned to the **Picture** property of components, to get the name of the figure, double-click on it, that its name will be transferred to the position of the cursor in the text box code.



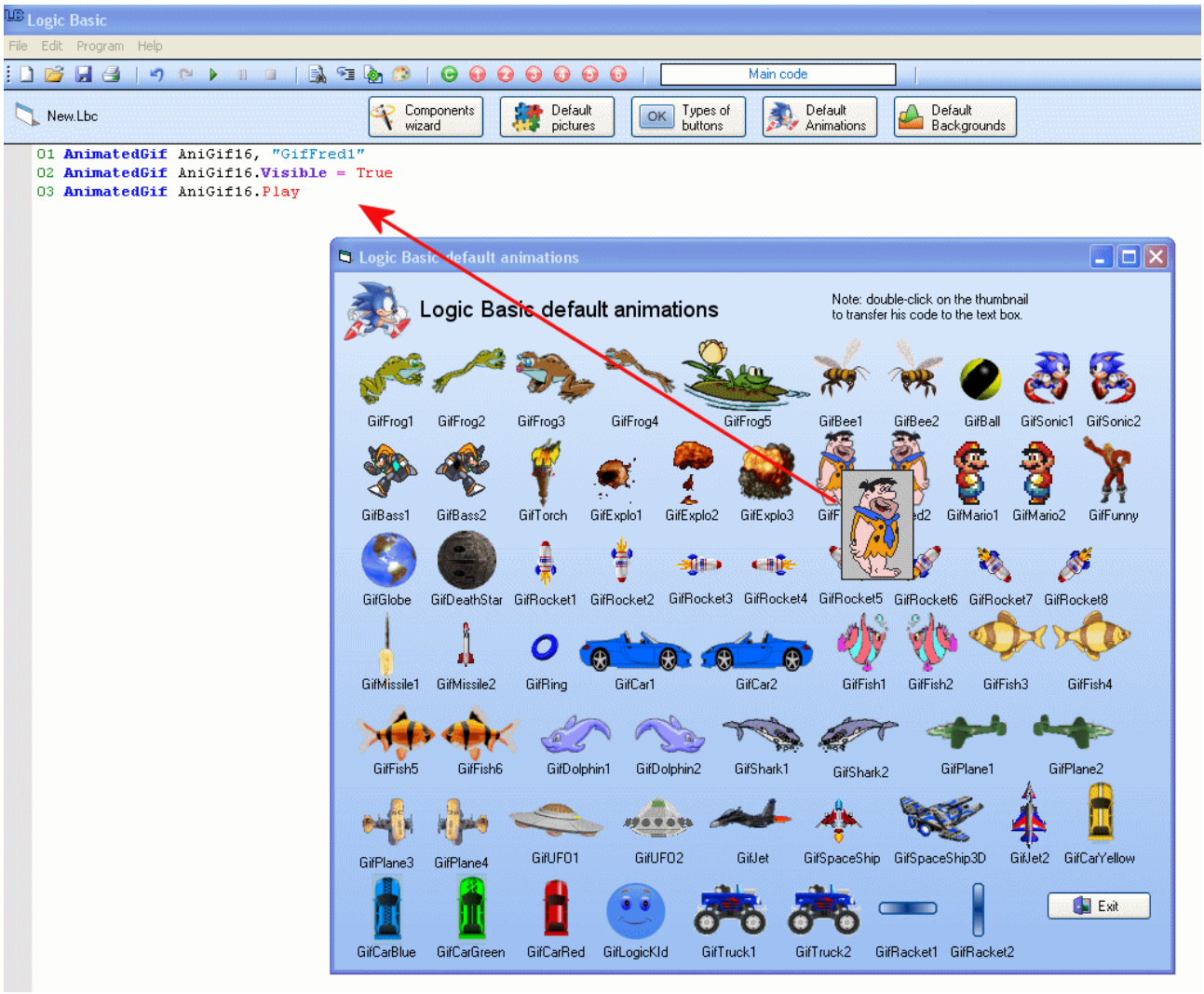
1.10 Types of Buttons

You can create buttons of various predefined types by clicking the "**Types of Buttons**" button and clicking the type of button you want to create. Once this is done, the button code will be transferred to the cursor position in the text box, and you can change the button position (line, column) by changing the third and fourth arguments.



1.11 Default Animations

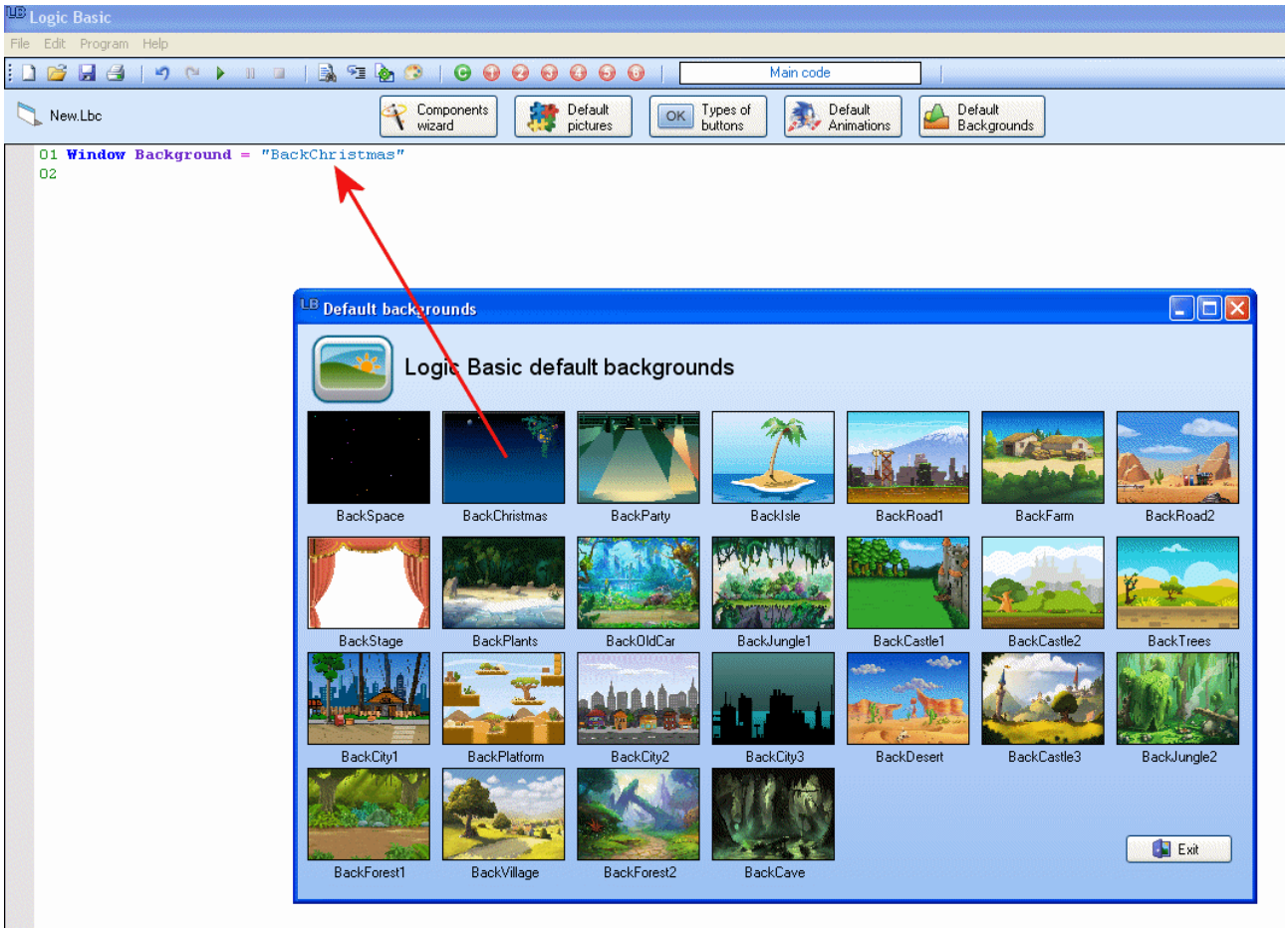
Logic Basic provides several standard animations that can be used in your programs, especially games, and you can easily create an animation with basic properties by clicking the "**Default Animations**" button, choosing an animation and double-clicking on it, then the animation code will be transferred to the text box.



Then you can change the properties and methods of the animation to change the position, size, and so on.

1.12 Default Backgrounds

Logic Basic also offers several images that can be used as background of the windows of your programs, to generate the code corresponding to a background click on the button "**Default Backgrounds**", choose an image and double-click on it.


Logic Basic
File Edit Program Help
Main code
New.Lbc
Components wizard
Default pictures
OK
Types of buttons
Default Animations
Default Backgrounds
01 Window Background = "BackChristmas"
02
Default backgrounds
Logic Basic default backgrounds
BackSpace
BackChristmas
BackParty
BackIsle
BackRoad1
BackFarm
BackRoad2
BackStage
BackPlants
BackOldCar
BackJungle1
BackCastle1
BackCastle2
BackTrees
BackCity1
BackPlatform
BackCity2
BackCity3
BackDesert
BackCastle3
BackJungle2
BackForest1
BackVillage
BackForest2
BackCave
Exit

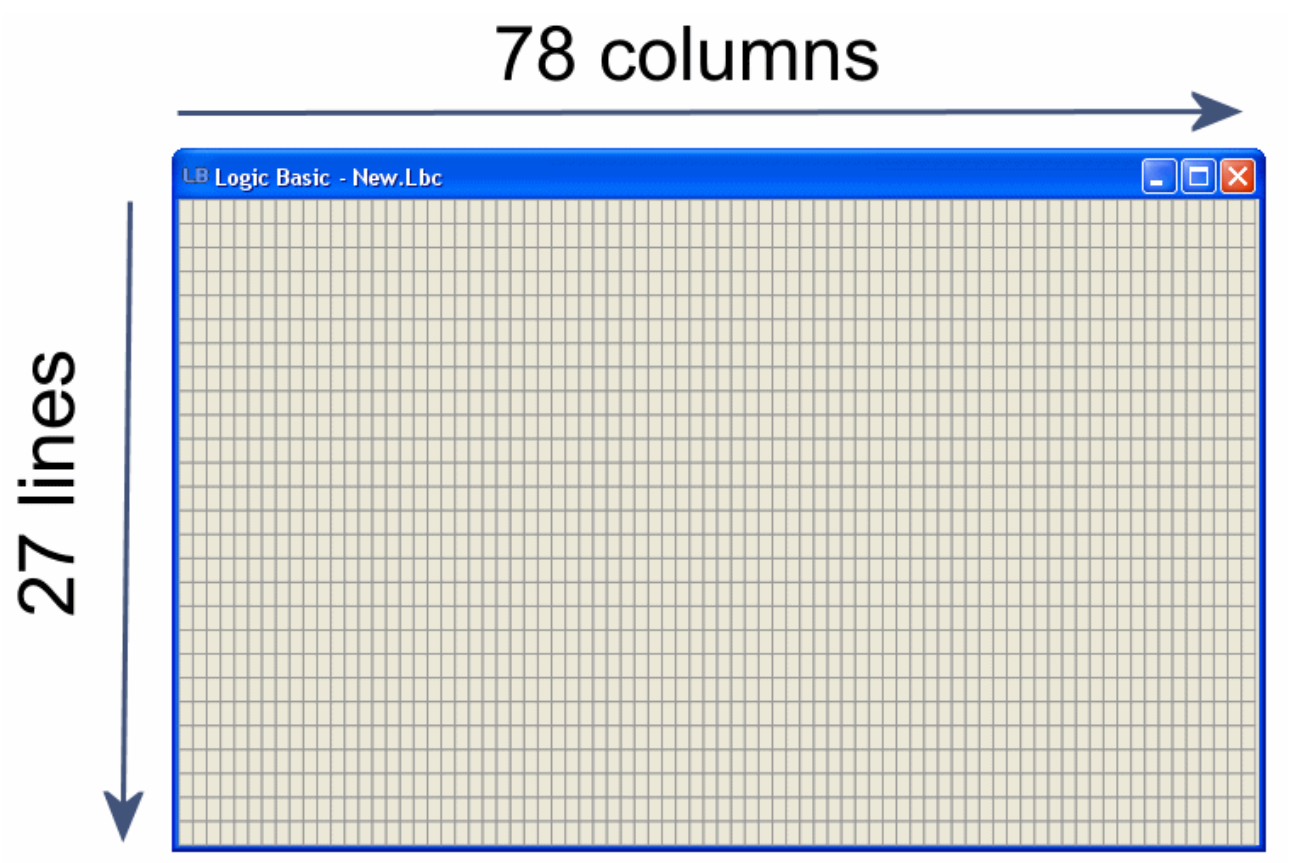
Chapter 2

Creating Logic Basic Applications

2.1 The Main Logic Basic Window

When you run a program in Logic Basic, the main window is activated to display the result of the first program commands. You can use only the main window, or new windows that can be turned on and off at any time to perform other routines.

To position a text, component, or graphic in the main window, you must tell LB the row and column to be placed. By default the main window has 27 rows by 78 columns, but this resolution can be changed in the **Window** command.



2.2 Writing a text in the main window

To write a text in the main window in a position defined by Line, Column, we first inform the desired position, and then the text to be written in the window:

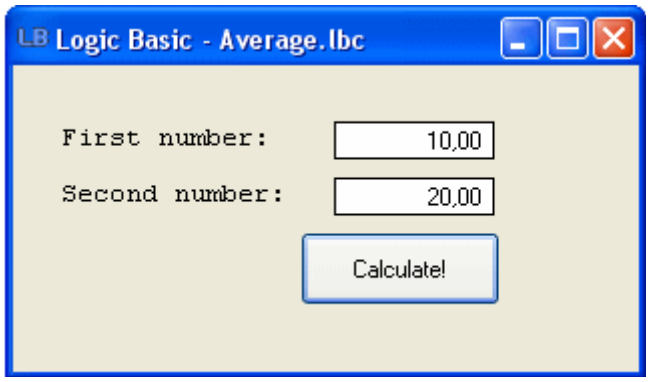
```
Position 11, 30
Write "Welcome to Logic Basic!"
```

Then press the "Run" button, or press F5. The result should be the phrase **Welcome to Logic Basic!** written in the center of the main window.

To close the window and return to the code text box, press F6.

2.3 Creating a small application in Logic Basic

Our first application will be a program to calculate the average of two numbers:



Generally the basic steps for creating an application in Logic Basic are as follows:

Step 1: Scale and configure the application window.

Step 2: Create the program interface.
Step 3: Write the code that will control the program.
So the first step is to define the size and position of the window, which will be in the center of the video monitor:

```
Window Size = 10, 40, Pos = Center, Center
```

With this configuration our window will have 10 lines by 40 columns and will be centered in the video monitor.

The second step is to create the program interface, which will write two sentences and create two Currency components and one Command Button:

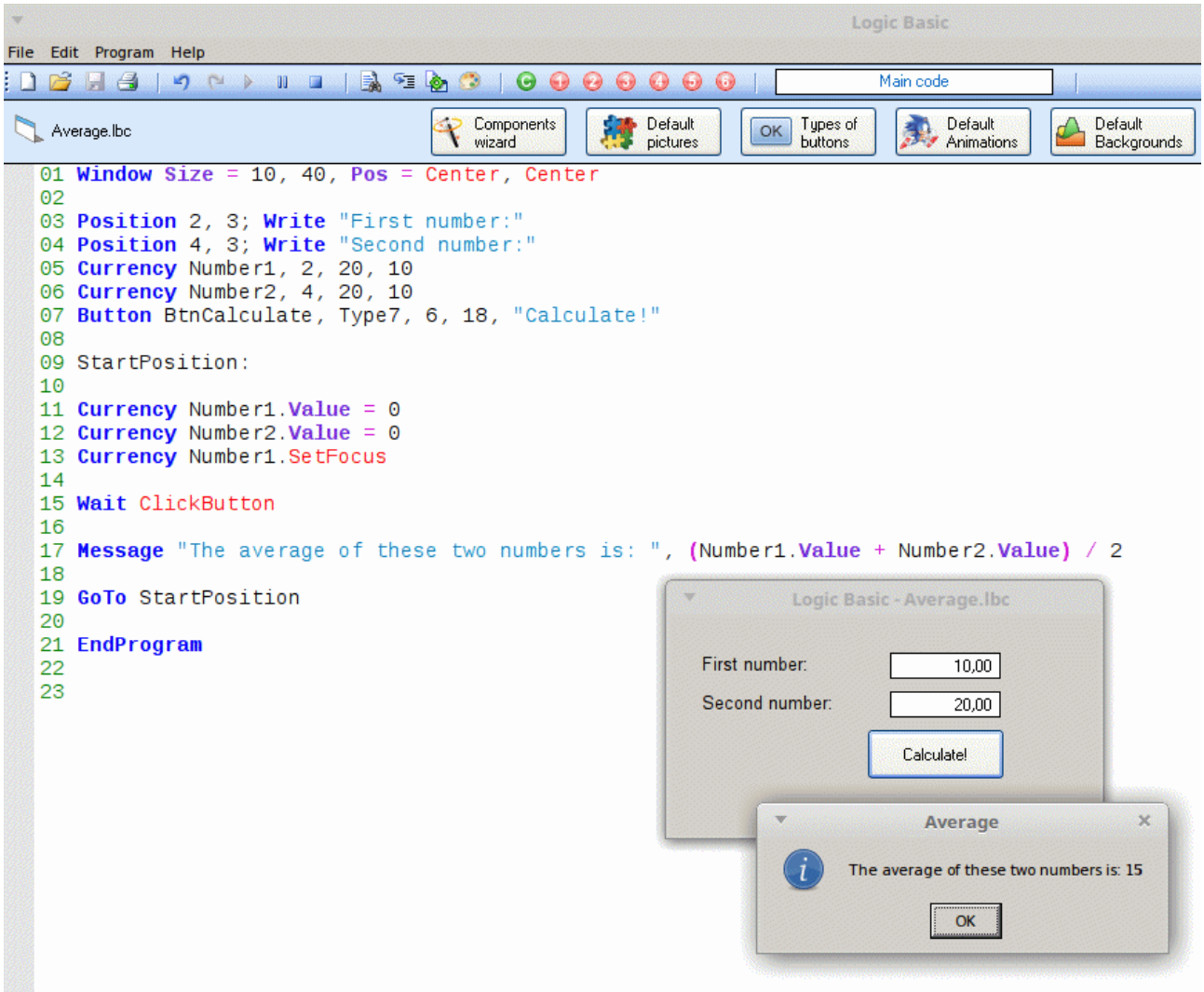
```
Position 2, 3; Write "First number:"
Position 4, 3; Write "Second number:"
Currency Number1, 2, 20, 10
Currency Number2, 4, 20, 10
Button BtnCalculate, Type7, 6, 18, "Calculate!"
```

The third step is to write the code that will control the program, in our case, start the variables, position the cursor in the first field, wait for a click on the button, and calculate the average of the two numbers:

```
StartPosition:

Currency Number1.Value = 0
Currency Number2.Value = 0
Currency Number1.SetFocus

Wait ClickButton

Message "The average of these two numbers is: ", (Number1.Value +
Number2.Value) / 2

GoTo StartPosition
```

EndProgram

In the code that controls the program, we put a label of name "StartPosition" to mark the beginning of the routine. Then we start the two numeric fields with the value equal to zero, and place the cursor in the first field. Then we call the command "Wait ClickButton" to wait for a click on the button. When the button is clicked a message will appear with the result of the average of the two numbers. Finally, with the command "GoTo StartPosition", we direct the execution of the program to the beginning of the routine, thus allowing a new calculation.



Chapter 3

Working with Variables

3.1 Variables Explanation

A variable is a user-supplied name to a location in the computer's memory, which can contain text, numbers, and characters, and can be modified at any time by the program.

In Logic Basic all variables must be declared before they are used, because the program needs to know the type of each variable to work with them correctly.

3.2 Declaring Variables

In other programming languages there are many types of variables, for example, numbers can be Byte, Small Integer, Integer, Long Integer, Float, Double Precision, and so on. In the old days it was very important to have many types of variables, because the old computers had little memory and disk space, and it was necessary to design the programs to save the maximum memory.

As one of the goals of Logic Basic is to simplify programming, and today's computers have lots of memory and disk space, all types of variables are summarized into three types: String, Integer, and Decimal.

To declare a variable you must type the word **Variable** or simply **Var**, and then the variable name and its type. If you do not enter the variable type, Logic Basic will assume that it's of type String. You can declare multiple variables on a single line by separating them with a comma, for example:

```
Variable Customer String, Age Integer, Salary Decimal
```

or simply

```
Var Customer String, Age Integer, Salary Decimal
```

To assign texts and numbers to variables, you must write the name of the variable, then the = operator (equal) and the text or value to be assigned, for example:

```
Customer = “Steven Spielberg”
Age = 70
Salary = 12345.67
```

Note that strings when they are assigned literally must be enclosed in quotation marks, while numeric values can not be enclosed in quotation marks. A number when enclosed in quotation marks will be recognized as a string, and when unquoted, will be recognized as a numeric value.

3.3 Rules for Variable Names

A variable name must contain at least one letter, may have only letters or a combination of letters, numbers, and underscores, and may contain letters, numbers, or underscores at the beginning of the name.

Can not contain spaces, quotation marks, operators, can not have the same name of the commands, functions, subroutines, Logic Basic keywords, and you should avoid putting these names inside the variable name.

3.4 Variable Data Types

String: Variables of type String can store a sequence of letters (texts), characters and numbers (numeric characters). To literally assign text or characters to a string, they must be enclosed in quotation marks:

```
Var S String

S = “Welcome to the Logic Basic!”
```

Integer: Integer variables store only integer numeric values, which can range from -2,147,483,648 to 2,147,483,647.

Decimal: Decimal variables can store double-precision floating-point numbers, as well integer numbers. Supports values that range in value from -1.79769313486231570E+308 through -4.94065645841246544E-324 for negative values and from 4.94065645841246544E-324 through 1.79769313486231570E+308 for positive values. Double-precision numbers store an approximation of a real number.

3.5 Global and Local Variables

Global variables are usually declared in the main code (see rules in the next topic) and can be accessed (used) in the main code, extensions, functions or subroutines. Local variables are declared inside a function and can only be used within it. It is possible to have local variables with the same name in different functions without causing conflicts.

It’s not a good programming technique you declare many global variables, especially in large programs, as they can cause conflicts and confusion in the execution of your program, for example, you can inadvertently change the value of a global variable in a subroutine,

and you do not see this in the main code of the program, in this case it will be difficult for you to detect the error.

Otherwise, when you declare a local variable, even if it has the same name as another variable created in the main code or inside another function, you are sure that any change in it will not affect the value of other variables.

3.6 Rules for Variables Declaration

Global variables are declared in any line of the main code, code extensions, subroutines or in the middle of functions.

Local variables are declared only in the first few lines of a function (not subroutines).

Example:

```
Variable X Integer, S String 'These variables are global

X = 7
S = "is a prime number"

Sum() ‘Executes the function

Write X, " ", S
Write "Ret = ", Ret

EndProgram

Function Sum()
Variable X Integer, Y Integer 'These variables are Local

X = 10; Y = 20

Variable Ret Integer 'These variable is global

Ret = X + Y
EndFunction
```

The result of the above program will be as follows:

7 is a prime number
Ret = 30

Note that the variable X has been declared globally and locally, and within the function the value 10 is assigned to that variable, and after the execution of the function, the value of the global variable X (which is equal to 7) has not changed.